International Journal of Robotics Research vol. 11, pp. 320-328, Aug. 1992

Recursive Formulation of Operational Space Control

K. Kreutz–Delgado

Electrical and Computer Engineering Department
University of California, San Diego
La Jolla, California 92093–0407
e-mail: kreutz@bhaskara.ucsd.edu

A. Jain, G. Rodriguez

Jet Propulsion Laboratory/California Institute of Technology
4800 Oak Grove Drive, Mail Stop 198–219
Pasadena, California 91109
e-mail: jain@telerobotics.jpl.nasa.gov

Abstract

A recently developed Spatial Operator Algebra approach to modeling and analysis of multibody robotic systems is used to develop O(n) recursive algorithms which compute the Operational Space mass matrix and the Operational Space coriolis/centrifugal and gravity terms of an n–link serial manipulator. These algorithms enable an O(n) recursive implementation of Operational Space Control.

1. Introduction

The Operational Space of a manipulator is defined by the configuration space of the end–effector (the standard cartesian task space). Operational Space Control (OSC) is an approach to manipulator control that focuses on the dynamical behavior of a serial rigid link manipulator as seen at the end–effector as it evolves in its Operational Space (Khatib 1983; Khatib 1985; Khatib 1987).

Let $y(0) \in R^6$ be a vector of generalized coordinates for the end–effector (Operational Space coordinates) and let $\beta(0) \in R^6$ be a generalized velocity vector for the end–effector (the Operational Space velocity). The argument "0" denotes the fact that our interest is with some point fixed with respect to the "tip" of the manipulator. The linear (but configuration dependent) kinematical relationship that exists between $\beta(0)$ and $\dot{y}(0)$, say

$$\beta(0) = \Pi\, \dot{y}(0) \ , \tag{1}$$

can always be used to restate expressions involving $\beta(0)$ and $\dot{\beta}(0)$ as forms involving $\dot{y}(0)$ and $\ddot{y}(0)$ (Khatib 83; Kreutz 89).

The associated Operational Space dynamics is

$$\Lambda(0)\dot{\beta}(0) + c(0) + g(0) = F(0) \tag{2}$$

where the relationship between the end–effector generalized spatial force, $F(0) \in R^6$, and the joint–level generalized force, T, is given by

$$T = J^* F(0) \ . \tag{3}$$

The term $c(0)$ gives the coriolis/centrifugal forces and $g(0)$ gives the gravity loading due to the manipulator being in a 1–g uniform gravitational field. The "*" denotes the adjoint operation and is equivalent to taking the transpose when J is given a matrix representation.

For a nonredundant (i.e. 6 dof) and nonsingular manipulator, $\beta(0)$ is a set of generalized velocities for the manipulator itself, and (2) describes the dynamical behavior of the entire arm. For a redundant manipulator, $\beta(0)$ cannot be a complete set of generalized velocities for the entire manipulator and, in this case, eq. (2) describes only how the end–effector responds dynamically to an applied Operational Space tip force $F(0)$.

In the OSC approach to manipulator control, a key step is to take $T = J^* F_c(0)$ where $F_c(0)$ is taken to be

$$F_c(0) = \Lambda(0)u + c(0) + g(0) \ . \tag{4}$$

Assuming that $\Lambda(0)$ is nonsingular, this results in decoupled and (almost) linearized end–effector dynamics of the form

$$\dot{\beta}(0) = u \ . \tag{5}$$

This behavior can be said to be almost linearized since the use of the kinematical relationship between $\dot{\beta}(0)$ and $\ddot{y}(0)$ (namely $\dot{\beta}(0) = \Pi\,\ddot{y}(0) + \dot{\Pi}\,\dot{y}(0)$, from (1)) can be used to obtain the decoupled and linearized behavior $\ddot{y}(0) = v$ by an appropriate choice of u (i.e., $u = \Pi\,v + \dot{\Pi}\,\dot{y}(0)$ (Khatib 1983; Kreutz 1989)). Of course, for the special case when $\dot{\beta}(0) = \ddot{y}(0)$, eq. (5) is directly equivalent to $\ddot{y}(0) = v = u$. Once the end–effector obeys the decoupled form $\ddot{y}(0) = v$, it is straightforward to perform stable control by the use of classical control techniques. (Control of (5) directly is discussed by (Luh, Walker, and Paul 1980) and (Wen and Kreutz-Delgado 1991).) The use of nonlinear feedback to force the end–effector to obey linear dynamics is known as "feedback linearization" or "exact linearization". From a mathematical perspective, the OSC approach is equivalent to other proposed feedback linearizing approaches such as Resolved Acceleration Control, Geometric Control, and Decoupled Control (Kreutz 1989).

The merit of the OSC appoach over alternative approaches to feedback linearization of the end–effector (such as the Resolved Acceleration Control approach of (Luh, Walker, and Paul 1980)) is that knowledge of the Operational Space dynamics can give key insights into the dynamical behavior of the end–effector under the influence of external forces. This is very important for applications involving contact between the end–effector and the environment, such as occurs in hybrid force/position control, or for applications involving artificial potential field forces for the purposes of end–effector collision avoidance (Khatib 1983; Khatib 1985; Khatib 1987; Khatib 1988). Unfortunately, a major difficulty of the OSC approach as it has been used in the past has been the need to have explicit analytical expressions for the Operational Space mass matrix $\Lambda(0)$, the Operational Space coriolis/centrifugal term $c(0)$, and the Operational Space gravity loading term $g(0)$. Such expressions can be quite complicated — for example, Armstrong, Khatib, and Burdick (1986) give analytical expressions for the *joint space* dynamical equations of motion of a PUMA 560 manipulator, expressions which are less intricate than those needed to describe the Operational Space dynamical behavior, and even these simpler equations amply demonstrate the complexity that can arise.

In (Rodriguez 1987; Rodriguez and Kreutz 1988; Rodriguez, Kreutz-Delgado, and Jain 1991; Rodriguez, Jain, and Kreutz-Delgado 1989; Jain 1991), a new approach to manipulator modeling and control, the Spatial Operator Algebra (SOA) framework for multibody system dynamics, has been developed. This framework allows even the most general time–varying closed–chain graph topology multibody robotic systems to be modeled and analyzed while keeping the complexity associated with such systems to manageable proportions. SOA provides a natural hierarchy of abstraction that enables high level analytic expressions to be easily manipulated and physically interpreted and from which efficient recursive algorithms are straightforward to produce. In this paper we will use the Spatial Operator Algebra to develop an $\mathrm{O}(n)$ recursive algorithm which implements OSC control, where n is the number of manipulator links. To do this, we will show how $\Lambda(0)$, $c(0)$, and $g(0)$ can each be generated by separate $\mathrm{O}(n)$ algorithms.[1] An $\mathrm{O}(n)$ algorithm for computing the Operational Space mass matrix $\Lambda(0)$ using the Spatial Operator Algebra can be found in (Rodriguez and Kreutz 1988). However, this paper is the first complete explicit statement and description of how to perform iterative OSC control using the Spatial Operator Algebra, including the recursive computation of the Operational Space gravity and coriolis/centrifugal terms.

To fill in all the steps implied by the abstract tools used here would require an entire and lengthy exposition of the Spatial Operator Algebra. Therefore, to preserve succinctness and to focus on the problem of developing recursive OSC control algorithms, the development of this paper has been kept intentionally terse. For further details on the use and nature of the Spatial Operator Algebra, the reader should consult the references (Rodriguez and Kreutz 1988; Rodriguez, Kreutz-Delgado, and Jain 1991; Rodriguez, Jain, and Kreutz-Delgado 1989; Jain 1991).

Recently, Lilly (1989) has independently developed via different means (by use of a physically motivated link–to–link inductive argument) an $\mathrm{O}(n)$ algorithm to produce $\Lambda(0)$ that can be shown to be equivalent to the one derived here. Since we do not emphasize implementation details

[1]It should be noted that, as is standard in the robotics literature (see, e.g., (Luh, Walker, and Paul 1980)), the term "recursive" is taken to mean "iterative" in this paper, and the two terms are used synonomously

of the iterative algorithms developed in this paper, the reader is encouraged to read Lilly (1989) where implementation issues as they pertain to the $\Lambda(0)$ algorithm are dealt with in some detail. In the work of Lilly (1989) and Lilly and Orin (1990) can also be found an efficient approximate method for computing $\Lambda(0)$ iteratively.

2. Operational Space Dynamics

In this section we give expressions for the Operational Space quantities $\Lambda(0)$, $c(0)$, and $g(0)$ in terms of the corresponding joint space quantities. For notational convenience, we have suppressed the notational dependence of $\Lambda(0)$ and $g(0)$ on configuration, and the notational dependence of $c(0)$ on configuration and velocity.

The joint–space dynamical equation of motion for a fully actuated n–link serial manipulator can be written as

$$\mathcal{M}\ddot{\theta} + \mathcal{C} + \mathcal{G} = T \tag{6}$$

where θ denotes the joint space configuration variable. $\mathcal{M}$ is the joint space mass matrix, $\mathcal{C}$ is the coriolis/centrifugal forces term and $\mathcal{G}$ is the gravity loading term. T is the vector of joint forces. In general the degrees–of–freedom (dofs) of the manipulator can exceed n since the joints between each link may have more than one dof associated with them (Rodriguez, Jain, and Kreutz-Delgado 1989). In fact, there may be as many as a full 6 dofs between two adjacent links. (This automatically takes care of the mobile base case since the base can be viewed as link 1, with a full six dofs between it and the ground). However, for the puposes of this paper it will not cause any harm to take the joints to be single degree–of–freedom joints, and this fact will be assumed henceforth. Thus, the n–link manipulator is assumed to have n fully actuated single degree-of-freedom joints so that $\ddot{\theta} \in R^n$ and $T \in R^n$ (and thus $\mathcal{M} \in R^{n\times n}$). Despite this restriction to an n degree-of-freedom manipulator, the results developed in this paper trivially extend to the multiple–dof joint case, including the case when the joint velocities and accelerations $\dot{\theta}$ and $\ddot{\theta}$ are replaced by generalized joint velocities v_θ and $\dot{v}_\theta$ where there is a (configuration dependent) linear relationship between $\dot{\theta}$ and v_θ (Rodriguez, Jain, and Kreutz-Delgado 1989).

The relationship between the end–effector generalized velocity $\beta(0)$ and the joint velocity $\dot{\theta}$ is given by

$$\beta(0) = J\dot{\theta}\ , \tag{7}$$

which implies that

$$\dot{\beta}(0) = J\ddot{\theta} + \dot{J}\dot{\theta}\ . \tag{8}$$

Generally, from among the many possibilities for $\beta(0)$, it is convenient to work with the end–effector generalized velocity defined by

$$\beta(0) \triangleq \begin{pmatrix} \omega(0) \\ \dot{x}(0) \end{pmatrix}\ , \tag{9}$$

where $\omega(0)$ and $\dot{x}(0)$ are the angular and linear velocity of the end–effector. When $\beta(0)$ is defined by (9), equation (7) gives the standard jacobian relationship between end–effector and joint velocities to be found in robotics textbooks (e.g., Craig 1989) and the corresponding Jacobian is called the "natural" or "standard" Jacobian (Khatib 1983). This important special case, where $\beta(0)$ is given by (9) and J is the standard Jacobian, is assumed to hold for the remainder of this paper. (Note, however, that the developments given below in this section are independent of this choice.)

We will now express $\Lambda(0)$, $c(0)$, and $g(0)$ of eq. (2) in terms of the joint level quantities $\mathcal{M}$, $\mathcal{C}$ and $\mathcal{G}$ of eq. (6), and the standard Jacobian, J, of (7). ¿From (6), we obtain

$$\ddot{\theta} + \mathcal{M}^{-1}(\mathcal{C} + \mathcal{G}) = \mathcal{M}^{-1}T$$

which with (8) becomes

$$\dot{\beta}(0) + J\mathcal{M}^{-1}(\mathcal{C} + \mathcal{G}) - \dot{J}\dot{\theta} = J\mathcal{M}^{-1}T \ . \tag{10}$$

Define

$$\Omega(0) = J\mathcal{M}^{-1}J^* \tag{11}$$

and take $\Lambda(0)$ to be

$$\Lambda(0) = \Omega(0)^{-1} = (J\mathcal{M}^{-1}J^*)^{-1} \ . \tag{12}$$

Premultipying (10) by $\Lambda(0)$ results in

$$\Lambda(0)\dot{\beta}(0) + \Lambda(0)(J\mathcal{M}^{-1}(\mathcal{C} + \mathcal{G}) - \dot{J}\dot{\theta}) = \Lambda(0)J\mathcal{M}^{-1}T = F(0) \tag{13}$$

using the fact that

$$\Omega(0)F(0) = (J\mathcal{M}^{-1}J^*)F(0) = J\mathcal{M}^{-1}T \ ,$$

where $F(0)$ is the Operational Space tip force in equations (2) and (3).

Comparison of (6) with (13), and distinguishing between velocity dependent and non–velocity dependent terms results in the identifications

$$c(0) = \Lambda(0)(J\mathcal{M}^{-1}\mathcal{C} - \dot{J}\dot{\theta}) \tag{14}$$

$$g(0) = \Lambda(0)J\mathcal{M}^{-1}\mathcal{G} \ . \tag{15}$$

Equations (12), (14) and (15) give the desired relationships between the Operational Space and joint space quantities.

Note that with these relationships the feedback linearizing controller (4) can be written as

$$T = J^*\Lambda(0)(u + J\mathcal{M}^{-1}(\mathcal{C} + \mathcal{G}) - \dot{J}\dot{\theta}) \ . \tag{16}$$

As discussed by Kreutz (1989), an iterative implementation of (16) is provided by Resolved Acceleration Control. A recursive implementation of Operational Space Control in a manner distinct from Resolved Acceleration Control is defined in this paper as follows: a) Recursively compute $\Lambda(0)$, $c(0)$, and $g(0)$; b) Compute $F(0)$ from (4) and then recursively compute $T = J^*F(0)$.

Possible hybrid implementations of the feedback linearizing controller (16) are a possibility, of course. For example, one could implement the feedback linearizing controller recursively at sample rate using Resolved Acceration Control while at a slower rate $\Lambda(0)$, $c(0)$, and $g(0)$ can be recursively computed, using the algorithms developed in this paper, in order to monitor the configuration dependent Operational Space dynamical properties of the manipulator for the purposes of on–line trajectory and/or control–law modification. For instance, $\Lambda(0)$ might be monitored to ensure that proper dynamic manipulability is maintained (see, e.g., Khatib 1987).

3. Spatial Operator Algebra

3.1 Spatial Operators for an n–Link Manipulator

Fundamental linear spatial operators associated with an n–link, n–dof manipulator are H^*, M, and $\mathcal{E}_\phi$. They will be defined below. It should be noted that the link numbering scheme used in this paper increases as one moves from the tip of the manipulator to the base, so that link 1 is the outboard–most link and link $n+1$ denotes the base. Point "0" denotes the point fixed with respect to link 1 which defines the end effector location and orientation.

The fundamental operator H^* is defined by

$$H^* \triangleq \text{diag}[H^*(1), \cdots, H^*(n)] \in R^{6n \times n} \ , \tag{17}$$

where $H^*(k) \in R^6$ is the joint axis of the kth link (which is along the joint connecting links k and $k+1$). The form of $H^*(k)$ depends on the nature of the joint k. For Example, if joint k is a revolute joint,

$$H^*(k) = \begin{pmatrix} h(k) \\ 0 \end{pmatrix} \ ,$$

where $h(k) \in R^3$ is the unit 3-vector along the joint axis.

The fundamental operator $\mathcal{E}_\phi$ is defined by

$$\mathcal{E}_\phi \triangleq \begin{pmatrix} 0 & 0 & \cdots & 0 & 0 \\ \phi(2,1) & 0 & \cdots & 0 & 0 \\ 0 & \phi(3,2) & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & \phi(n,n-1) & 0 \end{pmatrix} \in R^{6n \times 6n} \ , \tag{18}$$

where $\phi(k,k-1) \in R^{6\times 6}$ is the *rigid–body* force transformation Jacobian that transforms a spatial force acting on link k at a reference point located on joint $k-1$ to an equivalent force acting at a reference point now located on joint k. Note that $\mathcal{E}_\phi$ has nonzero block entries only along its first block subdiagonal. The cross–link rigid–body force transformation Jacobian $\phi(k,k-1)$ has the form

$$\phi(k,k-1) = \begin{pmatrix} I & \tilde{\ell}(k,k-1) \\ 0 & I \end{pmatrix} \ ,$$

where $\ell(k,k-1) \in R^3$ is a 3-vector directed along link k from the reference point on joint k to the reference point on joint $k-1$ and $\tilde{\ell}$ is defined by $\tilde{\ell}x \stackrel{\Delta}{=} \ell \times x$ for any 3-vector x. A general interlink rigid–body force transformation Jacobian is defined by

$$\phi(i,j) \stackrel{\Delta}{=} \phi(i,i-1)\cdots\phi(j+1,j) = \begin{pmatrix} I & \tilde{\ell}(i,j) \\ 0 & I \end{pmatrix} \ , \tag{19}$$

where $\ell(i,j)$ is a 3-vector directed from joint i to joint j (from reference point to reference point) and $i \geq j$. $\phi(i,j)$ for $i \leq j$ is defined similarly. The rigid–body force transformation Jacobians $\phi(i,j)$ obey the group properties

$$\phi(i,i) = I \ , \qquad \phi(i,j)^{-1} = \phi(j,i) \ , \qquad \text{and} \qquad \phi(i,l)\phi(l,j) = \phi(i,j) \ .$$

The fundamental operator M is defined by

$$M \stackrel{\Delta}{=} \text{diag}[M(1),\cdots,M(n)] \in R^{6n\times 6n} \ , \tag{20}$$

where $M(k) \in R^{6\times 6}$, the spatial inertia of the k^{th} link, is given by

$$M(k) = \begin{pmatrix} \mathcal{I}(k) & m(k)\tilde{p}(k) \\ -m(k)\tilde{p}(k) & m(k)I \end{pmatrix} \ .$$

The quantities $m(k)$, $m(k)p(k)$, and $\mathcal{I}(k)$ are, respectively, the 0^{th}, 1^{st}, and 2^{nd} mass moments of link k about the reference point on joint k and $p(k) \in R^3$ is the location of the link k mass center with respect to the joint k reference point.

Two additional fundamental spatial operators associated with the manipulator are

$$\begin{aligned} B^* &\stackrel{\Delta}{=} [\phi^*(1,0),0,\cdots,0] \in R^{6\times 6n} && (21) \\ E &\stackrel{\Delta}{=} [0,\cdots,0,\phi(n+1,n)] \in R^{6\times 6n} \ . && (22) \end{aligned}$$

¿From the fundamental quantities (17), (18), and (20) we can obtain additional linear spatial operators associated with the manipulator. The important additional spatial operators ϕ, P, D, G, K, $\mathcal{E}_\psi$, and ψ are defined as follows:

$$\begin{aligned} \phi &= (I-\mathcal{E}_\phi)^{-1} \in R^{6n\times 6n} && (23) \\ P &= \mathcal{E}_\psi P \mathcal{E}_\psi^* + M = \text{diag}[P(1),\cdots,P(n)] \in R^{6n\times 6n} && (24) \\ D &= HPH^* = \text{diag}[D(1),\cdots,D(n)] \in R^{n\times n} && (25) \\ G &= PH^*D^{-1} = \text{diag}[G(1),\cdots,G(n)] \in R^{6n\times n} && (26) \\ K &= \mathcal{E}_\phi G \in R^{6n\times n} && (27) \\ \mathcal{E}_\psi &= \mathcal{E}_\phi(I-GH) \in R^{6n\times 6n} && (28) \\ \psi &= (I-\mathcal{E}_\psi)^{-1} \in R^{6n\times 6n} \ . && (29) \end{aligned}$$

Although the definitions (23)–(29) appear to be implicit, the block component–level interpretation of (24) corresponds to a link–to–link iterative definition of P (namely the iteration (37) given below) which allows all the quantities defined to be constructed in a sequential manner.

Note that P, D, and G are block diagonal. $P(k)$ is the articulated body inertia of the k^{th} link (Featherstone 1987) while the nonzero scalar

$$D(k) = H(k)P(k)H^*(k) \tag{30}$$

is the projection of $P(k)$ onto the k^{th} joint axis. Also note that

$$G(k) = P(k)H^*(k)D^{-1}(k) \in R^6 \ . \tag{31}$$

Similarly to $\mathcal{E}_\phi$, K only has non–zero block entries, $\{K(2,1), \cdots, K(n, n-1)\}$, along its first block subdiagonal where

$$K(k, k-1) = \phi(k, k-1)G(k-1) \in R^6 \ . \tag{32}$$

$\mathcal{E}_\psi$ also only has non–zero block entries, $\{\psi(2,1), \cdots, \psi(n, n-1)\}$, along its first block subdiagonal where

$$\psi(k, k-1) = \phi(k, k-1)[I - G(k-1)H(k-1)] \ . \tag{33}$$

$\psi(k, k-1)$ is the *articulated–body* cross–link force propagation Jacobian for link k.

Equation (33) holds for $k = 1 \cdots n+1$ and it is meaningful to define the additional spatial operator

$$W \overset{\Delta}{=} [0, \cdots, 0, \psi(n+1, n)] \ . \tag{34}$$

A general interlink articulated–body force transformation Jacobian is defined by

$$\psi(i,j) \overset{\Delta}{=} \psi(i, i-1) \cdots \psi(j+1, j) \ , \tag{35}$$

for $i \geq j$. The articulated–body force transformation Jacobians $\psi(i,j)$ obey the semigroup properties

$$\psi(i,i) = I \ , \qquad \text{and} \qquad \psi(i,l)\psi(l,j) = \psi(i,j) \ .$$

Since $\mathcal{E}_\phi$ and $\mathcal{E}_\psi$ are both nilpotent operators, the operators ϕ and ψ given by (23) and (29) can be shown to be both lower block triangular with 6×6 block elements. In particular ϕ and ψ are given by

$$\phi = \begin{pmatrix} I & 0 & \cdots & 0 \\ \phi(2,1) & I & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \phi(n,1) & \phi(n,2) & \cdots & I \end{pmatrix} \quad \text{and} \quad \psi = \begin{pmatrix} I & 0 & \cdots & 0 \\ \psi(2,1) & I & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \psi(n,1) & \psi(n,2) & \cdots & I \end{pmatrix} \ . \tag{36}$$

The action of the operator ϕ on a composite spatial quantity $y = \text{col}(y(1), \cdots y(n)) \in R^{6n}$ (typically a composite spatial force) to form the composite spatial quantity $z = \phi y$, $z = \text{col}(z(1), \cdots z(n)) \in R^{6n}$, is equivalent to the following tip–to–base recursion:

```
$z(0)$  =  $0$;
loop  for  $\mathbf{k = 1 \cdots n}$
        $z(k)$  =  $\phi(k, k-1) z(k-1) + y(k)$;
end  loop;
```

The action of the operator ψ results in a similar tip–to–base iteration.

Dually, the action of the operator ϕ^* on a composite spatial quantity $s = \text{col}(s(1), \cdots s(n)) \in R^{6n}$ (typically a composite spatial acceleration or velocity) to form the composite spatial quantity $\alpha = \phi^* s$, $\alpha = \text{col}(\alpha(1), \cdots \alpha(n)) \in R^{6n}$, is equivalent to the base–to–tip recursion

```
$\alpha(n+1)$  =  $0$;
loop  for  $\mathbf{k = n \cdots 1}$
        $\alpha(k)$  =  $\phi^*(k+1, k) \alpha(k+1) + s(k)$;
end  loop;
```

The action of the operator ψ^* results in a similar base–to–tip effect.

The quantities (24)–(28) can be computed by a tip–to–base discrete–step riccati–like iterative processing of the fundamental quantities (17)–(22). This iteration is implicit in equations (24)–(28) and is given by

```
$P(0)$  =  $0$ ,  $\psi(1,0) = \phi(1,0)$ ,  $K(1,0) = 0$;
loop  for  $\mathbf{k = 1 \cdots n}$
        $P(k)$  =  $\psi(k, k-1) P(k-1) \psi^*(k, k-1) + M(k)$;        (37)
        $D(k)$  =  $H(k) P(k) H^*(k)$;
        $G(k)$  =  $P(k) H^*(k) D^{-1}(k)$;
        $K(k+1, k)$  =  $\phi(k+1, k) G(k)$;
        $\psi(k+1, k)$  =  $\phi(k+1, k)[I - G(k) H(k)]$;
end  loop;
```

Note that $D(k)$, $G(k)$, $\psi(k+1, k)$, and $K(k+1, k)$, $k \geq 1$, can be computed as soon as $P(k)$ is available so that the iteration (37) depends upon available quantities.

3.2 Operator Formulated Dynamics

The dynamic and kinematic expressions describing the behavior of multibody systems can be shown to be made up of the linear spatial operators defined above. The spatial operators show how spatial quantities (such as forces, velocities, and accelerations) propagate between bodies.

For example, the adjoint of the standard Jacobian can be written as

$$J^* = H\phi B \ , \tag{38}$$

a factorization into spatial operators which shows how the effect of a manipulator tip force propagates from the tip to the base and then projects onto the joint axes. This reflects the fact that the action of the lower block triangular operator ϕ is equivalent to a tip–to–base propagation of link interaction forces. The statement that $T = J^* F_c(0) = H\phi B F_c(0)$ is thereby shown to be equivalent to the recursion

```
$f(0) = F_c(0);$
loop for $k = 1 \cdots n$
    $f(k) = \phi(k, k-1) f(k-1);$
    $T(k) = H(k) f(k)$
end loop;                                                    (39)
```

Dually, the action of the standard Jacobian $J = B^* \phi^* H^*$ on the joint velocities $\dot{\theta}$ to form the end–effector velocity $\beta(0)$,

$$\beta(0) = J\dot{\theta} = B^* \phi^* H^* \dot{\theta} \ , \tag{40}$$

can now be given the following interpretation in terms of the actions of the individual operator factors. First, the action of H^* on $\dot{\theta}$ results in relative link velocities. Next, the action of ϕ on the relative link velocities $H^* \dot{\theta}$ results in the composite vector of link absolute spatial velocities, $V = \mathrm{col}(V(1), \cdots, V(n))$, where $V(k) = \mathrm{col}(\omega(k), v(k)) \in R^6$ is the spatial velocity of link k with $\omega(k) \in R^3$ and $v(k) \in R^3$ the angular and linear velocity of link k respectively. Finally, the action of B^* on the link absolute velocities serves to propagate $V(1)$ to the reference point 0, producing the end–effector velocity $\beta(0)$. These actions are equivalent to the following base–to–tip recursion:

```
$V(n+1) = 0;$
loop for $k = n \cdots 1$
    $V(k) = \phi^*(k+1, k) V(k+1) + H^*(k) \dot{\theta}(k);$
end loop;                                                    (41)
$\beta(0) = \phi^*(1, 0) V(1)$ .
```

The initial condition $V(n+1) = 0$ reflects the fact that, with no loss of generality, the base is taken to be immobile in this paper.

Operator factorizations also exist for $\mathcal{M}$, $\mathcal{C}$, and $\mathcal{G}$ of (6):

$$\mathcal{M} = H\phi M \phi^* H^* \tag{42}$$

$$\mathcal{C} = H\phi(M\phi^* a + b) \tag{43}$$

$$\mathcal{G} = H\phi M \phi^* E^* a_g \tag{44}$$

$$\mathcal{C} + \mathcal{G} = H\phi(M\phi^*\bar{a} + b) \tag{45}$$

where $\bar{a} = a + E^* a_g$ and $a_g = \text{col}(0, 9.8 \cdot e_g)$ with e_g is a unit 3–vector opposite to the direction of uniform gravitational attraction. It is also true that

$$\dot{J}\dot{\theta} = B^*\phi^* a + a(0) \ . \tag{46}$$

The vectors $a = \text{col}(a(1), \cdots, a(n)) \in R^{6n}$, $a(0) \in R^6$, and $b = \text{col}(b(1), \cdots, b(n)) \in R^{6n}$ are quadratic in the link velocities and are readily computed given these velocities. The vector $b(k) \in R^6$ is the "bias spatial force" associated with link k and is given by

$$b(k) = \begin{pmatrix} \omega(k) \times \mathcal{I}(k)\,\omega(k) \\ m(k)\omega(k) \times [\omega(k) \times p(k)] \end{pmatrix} \ . \tag{47}$$

The vector $a(k) \in R^6$, $k = 0, \cdots, n$, is the "bias spatial acceleration associated with link k and its form depends upon the nature of joints. For an all-revolute manipulator, $a(k)$ is given by

$$a(k) = \begin{pmatrix} \omega(k+1) \times \omega(k) \\ \omega(k+1) \times [\omega(k+1) \times \ell(k+1, k)] \end{pmatrix} \ . \tag{48}$$

Note that the quantities a, $a(0)$, and b are computable once the manipulator link velocities V have been found from the O(n) base–to–tip iteration given by $V = \phi^* H^* \dot{\theta}$. This means that the link bias accelerations and forces can be computed during the recursion (41).

The manipulator mass operator factorization, eq. (42), is referred to as the "Newton–Euler Factorization". The operator interpretation of the Newton–Euler Factorization gives the equivalence between the Newton–Euler and the Euler–Lagrange formulations of multibody dynamics (Rodriguez, Kreutz-Delgado, and Jain 1991). An alternative mass operator factorization is given by the "Innovations Factorization"

$$\mathcal{M} = [I + H\phi K]D[I + H\phi K]^* \ . \tag{49}$$

Since it is also true that

$$[I + H\phi K]^{-1} = [I - H\psi K] \tag{50}$$

we have that $\mathcal{M}$ has the operator inversion

$$\mathcal{M}^{-1} = [I - H\psi K]^* D^{-1} [I - H\psi K] \ . \tag{51}$$

The operator interpretation of (51) immediately results in an O(n) iterative solution to the manipulator forward dynamics problem. The Innovations Factorization (49) and the corresponding operator inversion (51) are discussed in detail in (Rodriguez, Kreutz-Delgado, and Jain 1991; Rodriguez, Jain, and Kreutz-Delgado 1989; Jain 1991).

3.3 Operator Identities

An important identity is

$$\phi K H \psi = \psi K H \phi = \phi - \psi \tag{52}$$

which can be easily used to show that

$$[I - H\psi K]H\phi = H\psi \ . \tag{53}$$

It is also true that

$$\psi M \phi^* = P + \tilde{\psi} P + P \tilde{\phi}^* = \psi P + P \tilde{\phi}^* \ , \tag{54}$$

where $\tilde{\phi} \stackrel{\Delta}{=} \phi - I$, $\tilde{\psi} \stackrel{\Delta}{=} \psi - I$, and P is given by the tip–to–base recursive implied by (24) (namely (37)). Note that from the definition (23) it is easy to show that

$$\tilde{\phi} \stackrel{\Delta}{=} \phi - I = \phi \mathcal{E}_\phi = \mathcal{E}_\phi \phi \ . \tag{55}$$

A related result to (54) is that

$$\psi^* H^* D^{-1} H \psi = \Omega + \tilde{\psi}^* \Omega + \Omega \tilde{\psi} \ , \tag{56}$$

where

$$\Omega = \mathcal{E}_\psi^* \Omega \mathcal{E}_\psi + H^* D^{-1} H \ . \tag{57}$$

Proofs of the identities (52)–(57) can be found in (Rodriguez and Kreutz 1988; Rodriguez, Kreutz-Delgado, and Jain 1991; Rodriguez, Jain, and Kreutz-Delgado; Jain 1991).

Note that the term $H^* D^{-1} H$ in (57) is block diagonal. As a consequence Ω is block diagonal,

$$\Omega = \mathrm{diag}[\Omega(1), \cdots, \Omega(n)] \ .$$

Ω can be computed via a base–to–tip recursion that is implied by eq. (57). This recursion is given by

```
$\Omega(n+1) = 0;$
    loop for $\mathbf{k = n \cdots 1}$
        $\Omega(k) = \psi^*(k+1,k)\Omega(k+1)\psi(k+1,k) + H^*(k)D^{-1}(k)H(k);$        (58)
    end loop;
```

Note also that given the fundamental spatial operators (17), (18), and (20), two recursions are actually needed to compute Ω since P, D, and $\mathcal{E}_\psi$ must be first computed during a tip–to–base recursion corresponding to the iteration (37).

A final identity is given by

$$(I - H^*D^{-1}H\psi P)E^* = W^* \ . \tag{59}$$

To prove (59), note that

$$\psi P E^* = \mathrm{col}[0, \cdots, 0, P(n)\phi^*(n+1,n)]$$

from which it is straightforward to show that

$$(I - H^*D^{-1}H\psi P)E^* = \mathrm{col}[0, \cdots, 0, (I - H^*(n)G^*(n))\phi^*(n+1,n)] \ .$$

4. Recursive Operational Space Control

The operator factorization of the standard Jacobian, eq. (38), tell us how to iteratively compute $T = J^*F_c(0)$, given the end–effector force (4), via the tip–to–base iteration (39). It remains then to show how to iteratively compute $\Lambda(0)$, $c(0)$, and $g(0)$.

4.1 The Operational Space Mass Operator

Equations (11), (51), and (52) result in

$$\Omega(0) = B^*\psi^*H^*D^{-1}H\psi B$$

which with (56) immediately gives

$$\Omega(0) = B^*\Omega B \ , \tag{60}$$

since $B^*\Omega\tilde{\psi} = 0$. The complete recursive algorithm for the computation of $\Lambda(0)$ is given by the combination of equations (57), (60), and (12):

$$\Lambda(0) = \Omega(0)^{-1} \ , \quad \Omega(0) = B^*\Omega B \ , \quad \Omega = \mathcal{E}_\psi^*\Omega\mathcal{E}_\psi + H^*D^{-1}H \ . \tag{61}$$

Note that since Ω is block diagonal, $\Omega(0) = \phi^*(1,0)\Omega(1)\phi(1,0)$. Therefore, computing $\Lambda(0) = \Omega(0)^{-1}$ via the recursive algorithm implied by (61) requires $\mathrm{O}(n)$ computations since n iterations are needed to produce $\Omega(1)$ from the recursion (58) and there is a flat cost associated with inverting the 6×6 matrix $\Omega(0)$. Based on the comments following (58), it is evident that the computation of $\Lambda(0)$ from the fundamental quantities (17), (18), and (20) actually requires two interative sweeps across the manipulator with a total operations count that is $\mathrm{O}(n)$.

4.2 The Operational Space Coriolis/Centrifugal Term

Note that equations (14), (38), (51), (46), and (53) taken together result in

$$c(0) = \Lambda(0)\left\{B^*\psi^*H^*D^{-1}H\psi(M\phi^*a + b) - B^*\phi^*a - a(0)\right\} \ . \tag{62}$$

¿From (54) and (52),

$$\begin{aligned}\psi^* H^* D^{-1} H \psi M \phi^* &= \psi^* H^* D^{-1} H(\psi P + P\tilde{\phi}^*) \\ &= \psi^* H^* D^{-1} H \psi P + \psi^* H^* G^* \tilde{\phi}^* \\ &= \psi^* H^* D^{-1} H \psi P + \psi^* H^* K^* \phi^* \\ &= \psi^* H^* D^{-1} H \psi P + \phi^* - \psi^* \ ,\end{aligned} \tag{63}$$

using (55) and (27).

Equations (62) and (63) together yield

$$c(0) = \Lambda(0) \left\{ B^* \psi^* [H^* D^{-1} H \psi (Pa + b) - a] - a(0) \right\} \ . \tag{64}$$

Equation (56) can be applied to (64) to produce alternative forms of $c(0)$ but our focus here will be only on (64). For purposes of computing numerical values of $c(0)$, it is preferable to work with the equivalent form

$$\Omega(0) c(0) = B^* \psi^* [H^* D^{-1} H \psi (Pa + b) - a] - a(0) \ . \tag{65}$$

Standard numerical techniques can be used to solve (65) once numerical values exist for $\Omega(0)$ and the right hand side of (65). The recursive algorithm implied by (65) is

```
$z(0)$ = 0;
loop for $\mathbf{k = 1 \cdots n}$
    $z(k) = \psi(k,k-1)z(k-1) + P(k)a(k) + b(k);$          (66)
    $\xi(k) = H^*(k)D^{-1}(k)H(k)z(k) - a(k);$
end loop;

$\eta(n+1)$ = 0;
loop for $\mathbf{k = n \cdots 1}$
    $\eta(k) = \psi^*(k+1,k)\eta(k+1) + \xi(k);$            (67)
end loop;
$\eta(0) = \phi^*(1,0)\eta(1);$

SOLVE: $\Omega(0)c(0) = \eta(0) - a(0)$ .                   (68)
```

The cost associated with computing numerical values for $c(0)$ from (65) (equivalently, from (66)–(68)) is O(n). Note that during the O(n) tip–to–base iterative sweep represented by the operator ψ, the quantities P, and D can be also computed since (37) and (66) can are both tip–to–base recursions. During the subsequent O(n) base–to–tip sweep implied by the operators ψ^* and ϕ^*, the quantitiy $\Omega(0)$ and the right hand side of (72) can be simultaneously computed (that is, the base–to–tip iterations (58) and (67) and be performed simultaneously), after which $c(0)$ can be solved for using a flat–cost efficient method.

In summary, to compute $c(0)$ by the algorithm (65) (given the velocity dependent terms a, $a(0)$, and b) requires two sweeps along the manipulator (a tip–to–base iteration followed by a base–to–tip iteration) and $\mathrm{O}(n)$ operations. Another base–to–tip $\mathrm{O}(n)$ iterative sweep is needed to first compute the velocity dependent quantities a, $a(0)$, and b before algorithm (65) can be applied. Thus, most generally, three iterative $\mathrm{O}(n)$ sweeps across the manipulator are needed to compute $c(0)$. Note, however, that by using (slightly) delayed velocity dependent terms, only two sweeps are needed since the velocity dependent terms can be computed during the second sweep of (65) and then used during the first sweep at the next invocation of algorithm (65).

4.3 The Operational Space Gravity Term

Similarly to the development in the previous subsection, equations (15), (38), (51), and (53) result in

$$g(0) = \Lambda(0) B^* \psi^* H^* D^{-1} H \psi M \phi^* E^* a_g \ . \tag{69}$$

Equations (63) and (69) then give

$$g(0) = \Lambda(0) B^* \left\{ \phi^* - \psi^* (I - H^* D^{-1} H \psi P) \right\} E^* a_g \ , \tag{70}$$

which with identity (59) results in

$$g(0) = \Lambda(0) B^* (\phi^* E^* - \psi^* W^*) a_g \ . \tag{71}$$

An equivalent statement, appropriate for obtaining numerical values of $g(0)$, is

$$\Omega(0) g(0) = B^* (\phi^* E^* - \psi^* W^*) a_g \ . \tag{72}$$

The recursive equivalent to (72) is given by

```
$\alpha(n+1) = a_g$ ,   $\eta(n+1) = a_g$;
    loop for $\mathbf{k = n \cdots 1}$
        $\alpha(k) = \phi^*(k+1,k)\alpha(k+1)$;
        $\eta(k) = \psi^*(k+1,k)\eta(k+1)$;                    (73)
    end loop;

SOLVE:  $\Omega(0)g(0) = \phi^*(1,0)(\alpha(1) - \eta(1))$ .   (74)
```

Note that there is no need to process velocity dependent terms in order to compute $g(0)$. Therefore, to compute $g(0)$ from the fundamental quantities (17), (18), (20), (21), and (22) requires no more than two $\mathrm{O}(n)$ iterative sweeps along the manipulator. A tip–to–base sweep to compute the quantities (23)–(29) and (34), followed by a base–to–tip sweep to simultaneously compute $\Omega(0)$ from (61) while performing the iteration implied by (72).

Finally, note that if only the sum $c(0) + g(0)$ is needed, rather than the separate quantities $c(0)$ and $g(0)$, then there is no need to implement algorithm (72) at all since, with a slight modification, algorithm (65) alone can be used to obtain $c(0) + g(0)$. Comparison of (43) to (45) and a reconsideration of the derivation of (65) shows that $c(0)+g(0)$ can be obtained from algorithm (65) (equivalently, from the iterations (66) and (67)) by use of the reassignment $a \to \bar{a} = a + E^* a_g$. This corresponds to the simple reassignment $a(n) \to a(n) + \phi^*(n+1, n)a_g$, showing that the iteration (66) is slightly modified only when $k = n$, while the iteration (67) is not affected at all.

5. Conclusions

The algorithm (61) enables $\Omega(0)$ and $\Lambda(0) = \Omega(0)^{-1}$ to be computed via two $\mathrm{O}(n)$ iterative sweeps of the manipulator. Algorithm (72) allows $g(0)$ to also be computed via two $\mathrm{O}(n)$ iterative sweeps across the manipulator. Assuming the availability of the velocity dependent terms a, $a(0)$, and b, algorithm (65) computes the quantity $c(0)$ (or, as discussed at the end of Subsection 4.3, the quantity $c(0) + g(0)$) at the cost of two $\mathrm{O}(n)$ iterative sweeps of the manipulator. These algorithms can be run simultanously to produce $\Lambda(0)$, $c(0)$ $g(0)$, and $F_c(0)$ of (4) as the output of a two–sweep $\mathrm{O}(n)$ iterative procedure (assuming the availability of the velocity dependent terms). As discussed at the end of Subsection 4.2, generally an additional $\mathrm{O}(n)$ iterative sweep is needed to compute the velocity dependent terms, but if a slight time delay can be accepted in the values of these terms, the need for an additional distinct sweep can be avoided.

Once $F_c(0)$ is available a final $\mathrm{O}(n)$ tip–to–base iterative sweep is required to compute the joint–level control forces $T = J^* F_c(0)$ via the recursion (39). However, if slightly delayed values of $F_c(0)$ can be tolerated, this last sweep can be merged with the tip–to–base sweep of the $c(0)$ algorithm (65). The overall Operational Space Control algorithm therefore requires at most four distinct sweeps across the manipulator. As few as two distinct sweeps (keeping the distinction between base-to-tip and tip-to-base sweeps) will suffice when delayed values can be tolerated.

Acknowledgement

The work reported here has been partially performed at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. K. Kreutz–Delgado was also supported by NSF Presidential Young Investigator Award No. IRI-9057631 and California Space Institute grant No. CS–22–90.

References

[1] Armstrong, B., Khatib, O., and Burdick, J. 1986 (April 7-10 1986, San Fransico). "The Explicit Dynamic Model and Inertial Parameters of the PUMA 560 Arm", *Proc. 1986 International Conf. Robotics and Automation.* Washington: IEEE Computer Society Press, pp. 510-518.

[2] Craig, J.J. 1989. *Introduction to Robotics*, 2^{nd} Edition. Reading: Addison–Wesley.

[3] Featherstone, R. 1987. *Robot Dynamics Algorithms.* Boston: Kluwer.

[4] Jain, A. 1991. Unified Formulation of Dynamics for Serial Rigid Multibody Systems. *J. Guidance, Control and Dynamics.* 14(3):531-542.

[5] Khatib, O. 1983 (Dec. 15–20 1983, New Delhi). Dynamic Control of Manipulators in Operational Space. *Proc.* 6^{th} *CISM–IFToMM Congress on Theory of Machines and Mechanisms.*

[6] Khatib, O. 1985. The Operational Space Formulation in the Analysis, Design, and Control of Robot Manipulators. *Proc.* 3^{rd} *International Symposium Robotics Research.* Cambridge: MIT Press, pp. 103–110.

[7] Khatib, O. 1987. A Unified Approach for Motion and Force Control of Robot Manipulators: the Operational Space Formulation. *IEEE J. Robotics and Automation.* RA-3:43–53.

[8] Khatib, O. 1988. Object Manipulation in a Multi–Effector Robot System. *Proceedings* 4^{th} *International Symposium Robotics Research.* Cambridge: MIT Press, pp. 137–144.

[9] Kreutz, K. 1989. On Manipulator Control by Exact Linearization. *IEEE Trans. Automatic Control.* TAC-34(7):763–767.

[10] Lilly, K.W. 1989. *Efficient Dynamic Simulation of Multiple Chain Robotic Systems.* PH.D. thesis, Ohio State University, Department of Electrical Engineering.

[11] Lilly, K.W., and Orin, D.E. 1990 (May 13-18 1990, Cincinnati). "Efficient O(N) Computation of the Operational Space Mass Matrix", *Proc. 1990 International Conf. Robotics and Automation.* Washington: IEEE Computer Society Press, pp. 1014-1019.

[12] Luh, J.Y.S., Walker, M.W., and Paul, R.P.C. 1980. On–line Computational Scheme for Mechanical Manipulators. *ASME Journal of Dynamical Systems, Measurement, and Control.* 25:468–474.

[13] Rodriguez, G. 1987. Kalman Filtering, Smoothing, and Recursive Robot Arm Forward and Inverse Dynamics. *IEEE Journal of Robotics and Automation.* RA-3(6):624-639.

[14] Rodriguez, G., Jain, A., and Kreutz-Delgado, K. 1989. Spatial Operator Algebra Framework for Multibody System Dynamics. *J. Astronautical Sciences.* 40(1).

[15] Rodriguez, G., and Kreutz, K. 1988. *Recursive Mass Matrix Factorization and Inversion: An Operator Approach to Open– and Closed–Chain Multibody Dynamics.* Pasadena: NASA/Jet Propulsion Laboratory Publication No. 88-11.

[16] Rodriguez, G., Kreutz-Delgado, K., and Jain, A. 1991. A Spatial Operator Algebra for Manipulator Modeling and Control. *International Journal of Robotics Research.* 10(4):371–381.

[17] Wen, J. and Kreutz–Delgado, K. 1991. The Attitude Control Problem. *IEEE Transactions on Automatic Control.* 36(10):1148–1162.